International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 10, October - 2013

Review: Performance Evaluation of TCP Congestion Control Mechanisms Using Random-Way-Point Mobility Model

Rakesh K
Scholar (M.Tech)
The Oxford College of Engineering
Bangalore

Mrs. Kalaiselvi
Asst. Prof, Dept of ISE
The Oxford College of Engineering
Bangalore

Abstract

Transmission control protocol(TCP) is one of the core protocols of IP suite. TCP is extensively used by many applications as it provides a reliable, ordered, congestion controlled, flow controlled, and error checked delivery of packets. TCP's congestion control plays a crucial role in delivery of packets and reducing traffic in the network. Many congestion control mechanisms have been developed and proposed. The purpose of this paper is to make a comparative study of these mechanisms and determine the better one. The various congestion control mechanisms considered in this paper are TAHOE, RENO, NEWRENO, SACK, and VEGAS. These mechanisms are implemented in MANET using AODV as underlying network layer protocol. Simulation is performed by varying the number of connections in the network and observing the change in Quality of Service with respect to throughput, delay and packet delivery fraction. Random mobility model is considered to introduce unpredictability among the network nodes as nodes are in random motion.

Keywords: Congestion Control, NewReno, Reno, SACK, Tahoe, TCP, Vegas

1. Introduction

TCP is a part of TCP/IP combination used by Internet. TCP makes sure that the data is put in the right order and none of it is missing. The reliability in data delivery is achieved through the mechanism of packet retransmissions in case of data losses. TCP segments the data into smaller units called packets and sends it through the network, the receiver re-orders the received packets with respect to the header information. Thus TCP is a reliable connection-oriented end to end protocol. TCP uses a simple timeout mechanism to perform operations of retransmissions. A timer is started on sending a packet, if an ACK is not received within a predefined time interval then the timer initiates a packet retransmission as shown in fig 1. Due to retransmissions congestion occurs in the network increasing the traffic and delayed delivery of packets, which again increases retransmissions. Hence congestion control is one of the critical areas to be inspected to improve the efficiency of the network and obtain a faster delivery rate. The various congestion control mechanisms developed so far are Reno, NewReno, Tahoe, SACK and Vegas. These mechanisms are simulated in MANET environment using AODV protocol for performance evaluation.

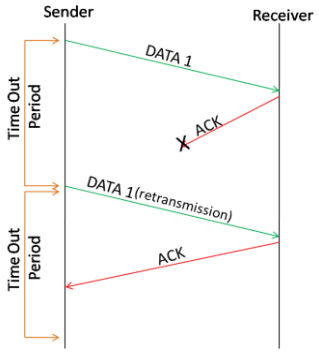

Fig 1 : TCP timeout mechanism for retransmission

MANET is a self configuring infrastructureless network of mobile devices connected by wireless media. Each device in MANET is free to move independently in any direction and therefore the links associated with each device change frequently. Each device forwards traffic unrelated to its own, this leads to congestion in network as a device is responsible for forwarding traffic of its neighbouring devices as well. AODV(Adhoc On-Demand Distance Vector) Routing

protocol is considered in simulation, as a dedicated path is established between the source and destination device with the use of route request and route reply messages as shown in fig 2, the sender broadcasts the RREQ(Route Request) message to all of its neighbours and these neighbours broadcast it to their neighbouring nodes and finally the message reaches the destination. The Destination sends RREP(Route Reply) only to the node from which it received the RREQ. Using AODV the traffic in the established path can be analysed and it is possible to determine whether congestion has occurred or not. To depict the mobility of devices in simulation, Random Waypoint Mobility model is considered.

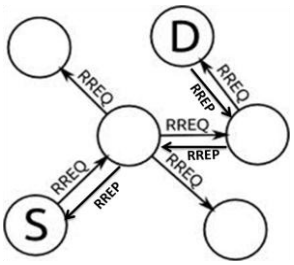

Fig 2 : RREQ(Route Request) and RREP (Route Reply) Mechanism of AODV

Random Waypoint Mobility model is a random model defined for the movement of mobile nodes. The mobile nodes move randomly and freely without restrictions. The model is defined such that each node pauses for a certain time, communicates with other nodes within its range and continues its random motion again. In real world the movement of the user is unknown and is random in nature. To evaluate the performance of protocols under such real time situations Random Waypoint Mobility model is used. We shall start the paper by taking a look at the various congestion control mechanism, compare the Quality of service of each mechanism by considering throughput, delay and packet delivery fraction using simulation and conclude with a better congestion control mechanism by a comparative analysis.

2. Congestion Control Algorithm

2.1 Slow Start and Congestion Avoidance

Once the connection establishment phase of TCP is completed, through three way handshake mechanism, the sender is unaware of how many packets it can send. To determine this slow start algorithm is implemented. The sender initially starts off by sending one packet, if it receives ACK then it doubles the number of outgoing packets and sends two. The process of increasing the number of packets exponentially is continued until the sender congestion window size reaches or exceeds a pre-defined threshold value. In order to avoid congestion, once the threshold value is reached the number of packets is increased linearly by a factor of one until it reaches congestion avoidance threshold. If any loss of packet or Dup Ack is received then the sender understands that congestion has occurred and has to reduce its transmission rate. The congestion avoidance threshold is reduced to half of the congestion window and congestion window is set to one as shown in fig 3. The sender follows the complete algorithm again starting by sending a single packet, increasing it exponentially and later linearly.

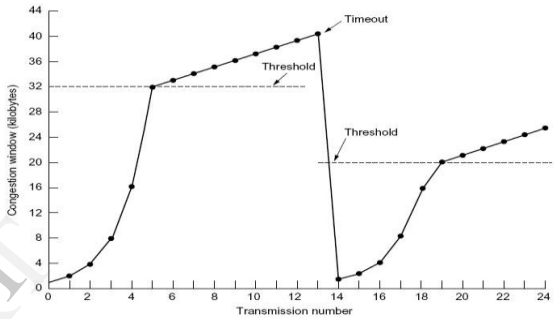

Fig 3 : Slow Start Mechanism

2.2 Fast Retransmit and Fast Recovery

The receiver is expected to send immediate dup Ack in case if it receives packets out of order. The dup Ack can be caused due to either dropped segments or due to re-ordering of data segments. In case of dropped segments, a dup Ack is sent for every received segment. The dup Ack carries the sequence number of the expected segment at the receiver. In such situations the sender implement " Fast Retransmit" algorithm. In this algorithm the sender doesn't wait till the timer expires, it retransmits the packet once it receives three dup Ack. TCP sender adopts Fast Recovery algorithm after a Fast Retransmit phase to forward data until it receives a non dup acknowledgement. Fig 4 depicts the Fast Recovery Mechanism and steps described below. The basic steps followed in the algorithm are :

1) Each time 3 dup Acks are received, Fast Retransmit mechanism is implemented followed by Fast Recovery.
2) In Fast Recovery the threshold value is set to half of the congestion window size and the congestion window is set to the new threshold value plus 3 Maximum segment sizes, owing to the 3 dup Ack received.
3) The congestion window size is increased linearly until a non dup Ack is received.

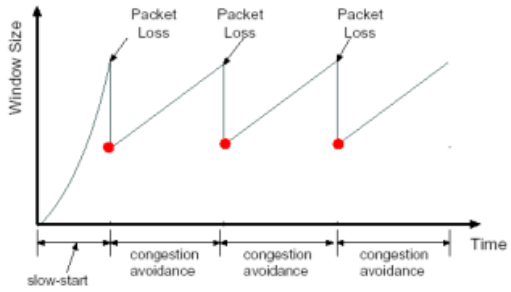

Fig 4 : Fast Recovery Mechanism

3. TCP Variants for Congestion Control

3.1 Tahoe

Tahoe is the TCP congestion control algorithm suggested by Van Jacobson. Tahoe uses timeout mechanism to determine packet loss. In most of the implementations it takes a longer duration due to coarse-grained timeouts. It implements the slow start algorithm for transmission purpose. For congestion avoidance it uses the mechanism of " Additive Increase Multiplicative Decrease". In this mechanism the sender sends the packets by increasing the number of packets exponentially until it receives Ack. If a time-out occurs then the congestion window is reduced to half of its current value. As this mechanism waits until the timer expires it introduces delay in channel. Delay in transmission leads to idleness of channel and thus under utilization of bandwidth.

3.2 Reno

This uses the mechanism of Fast Retransmit and Fast Recovery for congestion control. It performs well when the packet losses are small. In case of higher packet loss rate, it performs similar to Tahoe degrading the performance of the network and a time consuming process for detecting multiple loss. In case of multiple loss, the information about the loss of n+1 packet is received only after receiving Ack for the nth packet. If the congestion window size is set to a very small value, then we wouldn't receive the three dup Ack for a retransmit. The sender waits till the timer expires in such situation following the TCP Tahoe mechanisms.

3.3 NewReno

Considering the inability of Reno to handle multiple packet loss, respective modifications were made and implemented in NewReno. It differs from Reno in "Fast Recovery" phase. It doesn't exit the fast recovery phase until the sender receives Ack for all the data that were out-standing at the time it entered the Fast Recovery stage. The fast Retransmit phase remains the same. It takes one RTT to detect each packet loss. Although it solves the problem of detecting multiple packet loss, the solution is a time consuming process. It also implements Tahoe mechanism if the modified Recovery phase fails to detect packet loss at an earlier stage.

3.4 SACK

Selective Acknowledgement is a congestion control scheme implemented at the receiver side. It solves the problem of detection of multiple packet loss and retransmission of multiple lost packets per RTT. It implements slow-start and fast retransmit mechanism for congestion control. It inherits a timeout mechanism of Tahoe as back up, if the modified algorithm fails to detect packet loss. SACK requires that the segments need not be acknowledged every time it receives a packet but should be acknowledged selectively. Thus each Ack has a block which has information describing which segments were received and are acknowledged. In a network the receiver is often in a remote or a faraway place, hence it is difficult to implement this mechanism at the receiver site.

3.5 Vegas

This is a modification of Reno. It is built on the fact that " It is better to avoid congestion than to control it once it has occurred". It doesn't depend on packet loss as a sign of congestion. It implements modified slow start mechanism along with efficient time-out schedules to solve the problems faced by other variants. It still retains the other mechanisms of Tahoe and Reno, if the enhanced mechanism fails to detect packet loss. The major changes included in this are :

1.New Re- Transmission Mechanism

Vegas keeps track of when each segment was transmitted. It calculates an estimate of average RTT by recording the time it takes for the Ack to get back. When a dup Ack packet is received it checks if (current time-segment transmission time) > RTT, it retransmits the packet without waiting for the time-out period or for the 3 dup Acks.

2.Congestion Avoidance

Vegas doesn't consider loss of a segment as a signal to indicate that congestion has occurred. It determines congestion by reduction in sending rate as compared to the expected rate. Whenever the calculated rate is far from the expected rate, it increases the transmission rate exponentially to efficiently make use of the bandwidth. If the calculated rate is near to the expected rate then it reduces the transmission rate by increasing the number of packets linearly to avoid congestion.

3.Modified Slow Start

Vegas increases the number of packets exponentially only for alternate RTTs. Between that it calculates the sending throughput to the expected throughput. When the difference is above the threshold it exits slow start mechanism and enters the congestion avoidance mechanism.

4. Simulation Results

NS2 simulator was used to evaluate the performance of the different TCP variants for congestion control mechanisms with respect to throughput, delay and packet delivery ratio. A scenario generator script was used to generate random connections between the nodes. A CBR generator file was used to generate FTP traffic along with TCP Agents for TCP Tahoe, Reno, NewReno, SACK, Vegas accordingly.

Simulator Name	NS 2.34
Deployment Area	700*700
No. Of Nodes	50
No of Connections	10,20,30,40
Simulation Time	200sec
Traffic Type	FTP
TCP Agents	Tahoe,Reno,NewReno SACK, Vegas
Mobility Speed	10m/s
Packet Size	512
Routing Protocol	AODV
Queuing Model	DropTail
Mobility Model	Random Waypoint
Parameters	Throughput, Delay, Packet Delivery Ratio

4.1 Throughput v/s Number of Connections

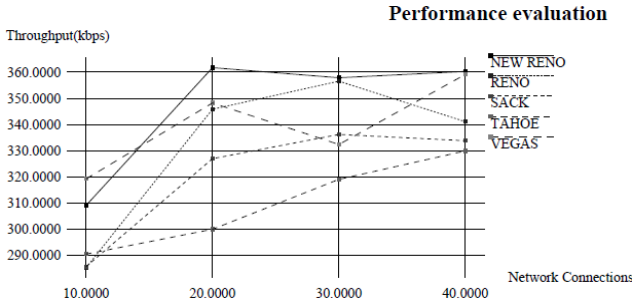

Fig 5 : Throughput Analysis

Fig 5 is a graph obtained by comparing the throughput against the number of connections in the network. It can be observed that the throughput of the variants vary by a small measure. The throughput degrades as number of connections increases due to collision of packets between various nodes.

4.2 Delay v/s Number of Connections

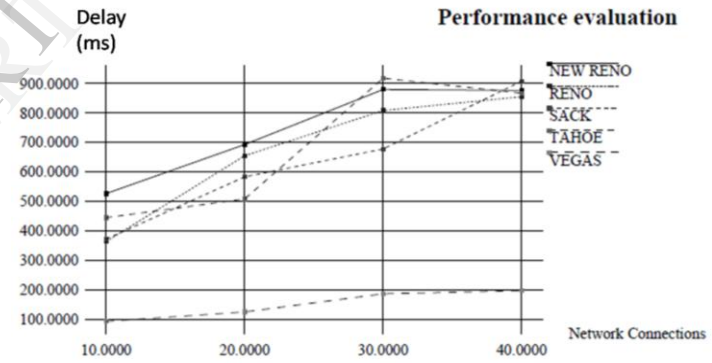

Fig 6 : Delay Analysis

Fig 6 is obtained by comparing end-to-end delay between the nodes. It can be seen that Vegas has the least delay when compared with the rest. Delay can occur due to a node participation in multiple connections, due to mobility, due to congestion or due to buffer overflow.

4.3 Packet Delivery Ratio vs Number of Connections

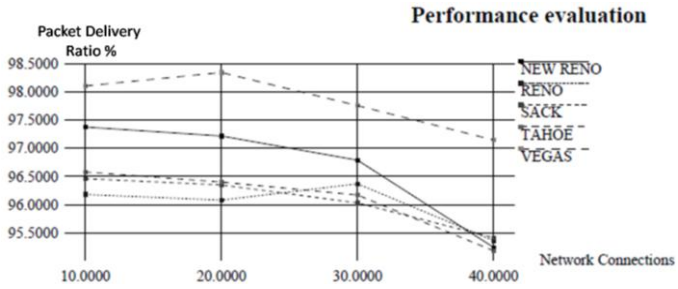

Fig 7 : Packet Delivery Ration Analysis

Fig 7 compares the packet delivery ratio of each variant. The pdf of Vegas remains above 97% for varying number of connections indicating that it delivers packet reliably whereas the other variants fall below 95 %.

5. Conclusion

We have successfully evaluated the five TCP variants namely Tahoe, Reno, NewReno, SACK and Vegas with respect to throughput, delay and packet delivery ratio to assess the quality of service provided by these congestion control mechanisms. Mobile Ad-Hoc Network was used along with Random Waypoint mobility model to approximate the simulation to depict a real world situation. The results obtained clearly state that TCP Vegas performs well when compared to all the other variants due to its enhanced mechanisms of retransmission and slow starts.

6. References

[1] RFC 2001 *" TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery"*.

[2] RFC 2581 *" TCP Congestion Control"*

[3] V. Jacobson. "Congestion Avoidance and Control".SIGCOMM Symposium no Communication Architecture and protocols.IJCN, Vol (2) 1988.

[4] L.S.Brakmo, L.L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet", IEEE Journal on Selected Areas in Communication, vol. 13[1995], (1465-1490)

[5]K.Fall, S.Floyd "Simulation Based Comparison of Tahoe, Reno and SACK TCP" International Journal of Advances in Engineering & Technology, Aug 2011.

[6] Poonam Tomar and Shweta Yadav "Enhanced Reliable TCP for congestion control with corruption control in MANETs" *Journal of Global Research in Computer Science* Volume 3, No. 4, April 2012.

[7] M. Jehan, Dr Radhamani and T KalaKumari "VEGAS: Better Performance than other TCP congestion control algorithms on MANETs" international journal of computer Networks (IJCN), Volume (3): Issue (2):2011.

[8] Mandakini Tayade and Sanjeev Sharma "Review of different TCP variants in ad hoc networks" IJEST ISSN: 0975-5462.

[9] Vishnu Kumar Sharma and Dr Sarita Singh Bhadauria "Performance Analysis on Mobile Agent Based Congestion Control using AODV routing protocol techniques with Hop by Hop algorithms for MANET" International Journal of Ad hoc, Sensor & Ubiquitous Computing (IJASUC) Vol.3, No.2, April 2012.

[10] Prof. S.A. Jain, Mr. Abhishek Bande, Mr. Gaurav Deshmukh, Mr. Yogesh Rade, Mr. Mahesh Sandhanshiv "An Improvement In Congestion Control Using Multipath Routing in Manet" International Journal of Engineering Research and Applications (IJERA) Vol. 2, Issue 3, May-Jun 2012,

[11] Ahmed Khurshid,Md.Humayun Kabir and Md.Anindya Tahsin Prodhan,"An improved tcp congestion control algorithm for wireless networks", IEEE, 2009.

[12] http://www.isi.edu/nsnam/ns/doc/

[13] www.wikipedia.com